Data-level parallelism

A question for the single instruction, multiple data (SIMD) architecture has always been just how wide a set of application has significant data-level parallelism. The answer is not only the matrix-oriented computation of scientific computing, but also the media-oriented image and sound processing. Moreover, since a single instruction can launch many data operations, SIMD is potentially more energy efficient than multiple instruction multiple data (MIMD), which needs to fetch and execute one instruction per data operation. These two reasons make SIMD attractive for Personal Mobile Devices (PMDs).

Finally, perhaps the biggest advantage of SIMD versus MIMD is that the programmer **continues to think sequentially**, yet **achieves parallel speedup** by having parallel data operations.

Variants of SIMD architectures

The three main variations of SIMD architectures are:

- Vector architectures
- Multimedia SIMD instruction set extensions
- Graphics Processing Units (GPUs)

The first variation, which predates the other two by more than thirty years, means essentially **pipelined execution of many data operations**. These vector architectures are easier to understand and to compile than other SIMD variations, but they were considered too expensive for microprocessors until recently.

The second variation borrows the SIMD name to mean basically **simultaneous parallel data operations** and is found in most instruction set architectures today that support **multimedia applications**: for x86 architectures, this started with *Multimedia Extensions* (MMX) in 1996, which were followed by several *Streaming SIMD Extensions* (SSE) versions in the next decade and continue to this day with *Advanced Vector Extensions* (AVX).

The third variation of SIMD comes from the GPU community, offering higher potential performance than is found in traditional multicore computers today. While GPUs share features with vector architectures, whey have their own distinguishing characteristics, in part due to the ecosystem in which they evolved. This environment has a system processor and system memory in addition to the

GPU and its graphics memory. In fact, to recognise those distinctions, the GPU community refers to this type of architecture as *heterogeneous.*